Friends and Circles — A Design Study for Contact Management in Egocentric Online Social Networks

Bo Gao and Bettina Berendt

Abstract Users in Egocentric Online Social Networks (EOSN) may share private information with the "wrong" friends. To mitigate this problem, we first designed an exploratory visualization for friend-grouping. We then conducted a user study, through which we found that, comparing Facebook smart lists, the hierarchal modularity-based communities were more helpful for users to make visibility decisions in online posting. We then compared the modularity-based algorithm (MOD) with another state-of-the-art community detection algorithm. The results showed that the ground-truth circles coincided more with the MOD-circles. We further extended MOD to produce overlapping circles and found even better results. Furthermore, informed by our user study, the research on social groups and information visualization theories in general, we developed a friend-exploration/grouping web application for Facebook users.

1 Introduction

An Online Social Network (OSN) today can hold hundreds of millions of users. Two years ago (2012), Facebook (www.facebook.com) has reached its "one billion users" mark [64]. Behavioural and sometimes very personal information of OSN users is uploaded and shared online daily, in large quantity and tremendous detail. While the availability of these data enables us to understand more about our societies, it also challenges us in effectively and efficiently processing large amount of information, and managing our online personal content.

Bo Gao
Department of Computer Science, KU Leuven, Celestijnenlaan 200A, 3001 Heverlee, Belgium.
e-mail: bo.gao@cs.kuleuven.be

Bettina Berendt
Department of Computer Science, KU Leuven, Celestijnenlaan 200A, 3001 Heverlee, Belgium.
e-mail: bettina.berendt@cs.kuleuven.be

in Kawash, J. (Ed.) (2014). Online Social Media Analysis and Visualization. Lecture Notes in Social Networks. Berlin etc.: Springer.

In the age of online social networking, "even as bloggers and networkers delve into their private experience, they communicate with their fellow humans in a shared festival of the self" [5]. Such phenomenon has raised concerns about privacy. For example, it was found that OSN users had demonstrated high privacy concerns while revealing great amounts of personal information [1]. It also has been quantitatively demonstrated that users' perceptions of audience size do not match reality, since not enough feedback is provided [8]. boyd showed that collapsed or ambiguous online contexts could lead to undesired disclosure of personal information [10]. Gürses extends the notion of privacy from confidentiality to access control and practice. The extended notion encompasses the solution space in which OSN users are empowered to re-negotiate the boundaries of information dissemination and construct their online identity based on a transparent system [30] .

In light of the recent privacy research, we become interested in the tools that can help OSN users gain insights into their own social networks, explore to reveal hidden patterns and control the flow of personal data shared with online friends. As previous studies have suggested [37, 48, 26], in order to manage personal information flow, it is important for users to categorize their online friends into groups, categories, circles, lists or communities[1], so that the user can post towards clearly specified audience. By "post", we mean the user's action of uploading or sharing digital information in OSN. We will also be using the term Egocentric Online Social Network (EOSN) to refer to a sub-network in an OSN, with the nodes representing people and the (directed or undirected) edges representing certain relationships among them. The network is centered on one user (as the ego), whose friends (as the alters) are directly linked to this user via edges. Edges usually also form among the friends.

As reported in 2011, the median number of friends of a Facebook user was 100 [57]. This number became 229 in 2013. For teens and people in their 20s, it was 400 or more [62]. To make sense of the increasingly complex online social networking data and manage online contacts, a user needs to deploy more sophisticated tactics (categorizing friends under different situations) than simple browsing and memorising. We started looking into visualization approaches to address this issue, as human visual system is highly parallel and pre-attentively sensitive to variations in visual stimuli, such as color, shape, positions, etc. [52]. With a carefully designed interactive visualization system, the user should be able to gain an overview of her network, explore the network to find novel patterns and easily construct groups of friends for different posting purposes.

The contributions of this chapter are: First, we documents a user study, which shows that, compared with Facebook smart lists, the hierarchical modularity-based circles (used in an exploratory fashion) are more supportive for users to make privacy-related visibility decisions in online posting. Second, we design a new form

[1] We use these words interchangeably throughout the paper. The words "group" and "category" are used more generically, "list" is often used in the context of Facebook and Twitter (www.twitter.com). We use "circle" more often in the context of Google+ (www.plus.google.com) and visualization. The word "community" is usually used in the context of community detection algorithms.

of interactive visualization to visualize hierarchically grouped items. The items can be individual friends a user has in her online social network. Third, we test two community detection algorithms on three egocentric social network datasets. The results show that the ground-truth circles coincide more with the modularity-based circles. Fourth, we extend the modularity-based algorithm to accommodate the overlapping nature of online social circles. The experimental results show that this approach is indeed better than the original modularity-based algorithm. Fifth, we develop a friend-exploration/grouping web application for Facebook users to explore their online social networks and create their customized friend-lists.

The structure of this chapter is as follows: Section 2 covers a set of existing tools for EOSN analysis and friend grouping. In Section 3, we analyze users' requirements, motivate and detail a new design of interactive visualization, named CircleTree. We then use it as the common interface to conduct a user study. The study compares users' behaviour in privacy decision-making based on two different friend-grouping mechanisms. In Section 4, we examine two community detection algorithms and discuss the nature of friend grouping in EOSN. We then propose an extended version of the modularity-based community detection algorithm to generate overlapping friend groups. In Section 5, with the introduction of the tool named FreeBu, we propose alternative views to supplement the earlier CircleTree visualization. We then identify the improvement points for the friend-exploration/grouping tool design. In Section 6, we conclude with a summary and an outlook on future work.

2 Related Work

We are interested in the tools that enable users to gain insights into their EOSN and/or construct friend groups. We describe a selective set of existing tools in Section 2.1, and discuss their relationships with our contributions in Section 2.2.

2.1 Existing Tools

PViz [38] is a tool that helps Facebook users understand their privacy settings. The tool is compared with existing policy comprehension tools on Facebook, namely Audience View and Custom Settings. It was shown that PViz was more effective for the users in comprehending privacy settings. Privacy Wizard [19] is a tool that can automatically predict the privacy preferences for a Facebook user based on her previous privacy-setting input, the result is also encouraging. Both tools employ Newman's Modularity-based community detection algorithm [46] to drive friend groups, which are then used for visualization (PViz) and prediction (Privacy Wizard) respectively.

NodeXL [53] is a general-purpose plugin that allows users to draw graphs by using a Microsoft Excel template. It implements various graph clustering algorithms, including modularity-based ones. It supports social network analysis, users can visualize their Facebook and Twitter graph data via an importer interface. The tool uses the Group-In-a-Box (GIB) feature [49] to help users delineate the clustering structure of the imported graphs. More specifically, the visual clusters are firstly formed in a graph layout. They are further constrained by being placed inside boxes whose sizes depend on the respective numbers of nodes. These boxes are then arranged by the squarified treemap algorithm [11]. The GIB layout is also used for multivariable grouping of the nodes based on their attributes. Some other general-purpose network-analysis tools are potentially useful for OSN users as well, such as Gephi [6], Cytoscape [51] and Tulip [3].

There also exist many small web applications that visualize OSN users' network data and allow simple interactions for exploration. Here we give two representative examples. Social Graph[2] is a Facebook application that shows a force-directed layout of the user's friend graph. The nodes are colored according to the detected communities based on Modularity [46]. The user can click on an individual friend (i.e. a node) to see the friend's profile photo along with three statistical numbers: the number of mutual friends she shares with that friend, the clustering coefficient of the friend and the clustering coefficient of the corresponding community. The user can further explore the graph layout by selecting a specific community from a drop-down list, so that only the members from that community are repositioned and displayed on the screen. Each community is labeled with the name of the friend in that community who has the highest clustering coefficient. The other example is InMaps[3]. It is a web application similar to Social Graph. It visualizes the user's network on LinkedIn (www.linkedin.com) with rather similar force-directed layout and modularity-based communities as well. Through InMaps, a LinkedIn user can zoom and pan to explore the map. The name labels of the friends are simultaneously brought to display upon zooming-in. We can also see that the nodes and labels are mapped with care to avoid overlapping, which makes the visualization more readable than Social Graph.

Personal Analytics for Facebook[4], as part of the Wolfram Alpha knowledge Engine (www.wolframalpha.com), is a state-of-the-art visual and textual analytic web application designed for Facebook users. It offers a wide range of analytics, including various friends' demographic reports, summaries of the user's logging-in, posting and sharing activities, etc. Another merit of this tool is that each analytic segment can be downloaded in different formats for other uses, such as spread sheet, image and vector graph.

Furthermore, current Social Networking Sites (SNS) provide mechanisms for users to create their own friend groups, such as the lists in Facebook and Twitter, the circles in Google+ and the groups in Weibo (www.weibo.com). But by large, users

[2] https://apps.facebook.com/socialgraph_fr_yl [Accessed on Nov 30, 2013]

[3] http://inmaps.linkedinlabs.com/network [Accessed on Nov 30, 2013]

[4] http://www.wolframalpha.com/input/?i=facebook+report [Accessed on Nov 30, 2013]

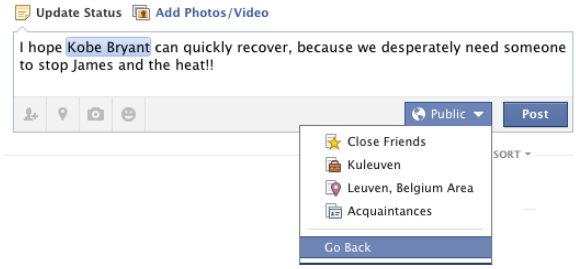

Fig. 1 A Facebook user can conveniently limit the visibility of her status by choosing one of the four lists, of which *Close Friends* and *Acquaintances* are the lists that the user manually defines, and *Kuleuven* and *Leuven, Belgium Area* are the automatically generated smart lists, based on the user's work and current city.

have to manually group friends, which tends to become unmanageable. We know one exception – Facebook "smart lists" — that can automatically generate friend lists. Facebook smart lists[5] provide users with an automatic grouping solution. The lists are generated based on the information about the user's education, work and current city. For example, if the user indicates Leuven as her current city, she will have a list with all of her friends who also indicate Leuven as their current city. The user can directly determine the audience of her posts by choosing one of the lists, including the smart lists. Figure 1 gives an example for status update.

2.2 The Tools' Relations to Our Contributions

Informed by PViz and Privacy Wizard, we find that one feature of Facebook's privacy control mechanism yet to be examined is the smart lists. In Section 3, we take the smart lists as baseline and investigate the roles that community detection algorithms and interactive visualization play in users' privacy decision-making process. We also note that PViz is for privacy-setting comprehension, Privacy Wizard is for privacy-setting configuration, both tools do not serve for the purpose of helping users create their own friend groups, e.g. Facebook friend lists. We built the friend-exploration/grouping tool (as detailed in Section 5) to facilitate this activity.

The GIB feature in NodeXL currently does not support hierarchical graph clustering and exploration. A hierarchy is difficult to visualize and interact with, because the semantics from different layers may compromise the readability of a set of visual clusters, especially when the leaf nodes are of main interest (e.g. the user' friends). In Section 3, we introduce a hierarchical exploratory visualization design. This design displays the grouping structure of the user's EOSN and maintains the emphasis on the leaf nodes of a hierarchy.

[5] https://www.facebook.com/help/204604196335128/ [Accessed on Nov 30, 2013]

Unlike Gephi, Cytoscape and Tulip, NodeXL offers its users connectivity to their Facebook accounts, so that they can easily analyze their ego-networks. Besides such connectivity, our tool also provides its users with a series of list-creation interfaces. The user-created friend lists can be submitted to their Facebook accounts. Moreover, NodeXL, Gephi, Cytoscape and Tulip are desktop applications/plugins that require installation. For NodeXL, the installation is conditioned on having Microsoft Excel in advance. Our tool is an online application that is easily accessible by a JavaScript-enabled browser. To facilitate user-defined friend-grouping, in Section 5, we elaborate the ways in which we improve the existing graph-layout visualizations, such as Social Graph and InMaps.

In Personal Analytics for Facebook, we find that the visualizations are fairly static as it follows the interaction syntax of a regular web page – that supports up/down scrolling and hyperlink clicking, but without zooming, panning and animation. Thus the action of inspecting individual objects in an overview visualization, e.g. foraging through the graph clusters, become problematic if the user has many friends. We consider our interactive visualization design to be complementary with respect to this.

Furthermore, we notice that all the aforementioned tools in Section 2.1 use or include modularity-based communities to approximate the user's social groups. Modularity maximization encourages mutually connected nodes to be put into the same community. While the broad adoption of this method is partly due to its popularity and software availability, another contributing factor seems to be that, among many other community detection methods, it produces the communities that best match the communities a user has in mind. Various studies have demonstrated useful applications of modularity-based community detection algorithms for social network analysis [46, 35]. We will also be using this method in our user study (Section 3). In Section 4, we examine this method in more detail and demonstrate its usefulness for EOSN friend-grouping. We also propose an extension of the modularity-based algorithm and show that it has a better performance.

3 A User Study on Circles for Visibility Decisions

This section consists of three parts: First, in relation to our user study, we discuss OSN users' need for friend-grouping tools (Section 3.1) and why we further choose to use hierarchical grouping (Section 3.2). Second, we examine the related work on visualizations for hierarchies (Section 3.3) and describe the CircleTree visualization that we have developed (Section 3.4). This visualization is then used in our user study. Third, we describe the participants, tasks and results of the user study (Section 3.5).

3.1 The Need for Grouping Tools

As discussed in Section 1 and 2, we know that the increasingly large amount of data produced by our online social networking activities has made it difficult for us to manage our personal information flow. Sharing certain information with the wrong people can cause awkwardness, embarrassment or even severe damage on the user. Therefore a tool is needed to inform OSN users and facilitate their privacy decision-making. More specifically, the user should be able to effectively determine which piece of her personal information is visible to which friend(s). But it would be a daunting task if the user goes through each individual online friend that she has one by one, and considers that friend's unique constellations of attributes and proclivities in order to make such a decision. In reality, informed by the research in social cognition [36], we know that people "prefer to construe others on the basis of the social categories to which they belong, categories for which a wealth of related material is believed to reside in long-term memory". Because of the limitations in human cognition and the challenges presented by a vast stimulus world (in our case – the online social networking environment, intertwined with the offline social life), a person naturally employs categorical thinking in order to simplify and structure the people she befriends [2, 36]. This description further provides support for the necessity of friend grouping [37, 48, 26].

We will be using the term Visibility Decision to refer to a user's binary decision on whether a post is visible to an individual friend in her EOSN. A post can be anything that a user uploads or shares in an OSN, e.g. a status update, a (re)tweet, a photo, a comment or an article shared, etc. Friend grouping can facilitate users' visibility decisions. The user decides the visibility of a post directly based on friend groups rather than individuals. In other words, when the user sees a group, assuming her previous familiarity with the group, she can skip the serial browsing that examines individual friends in this group, and determine the visibilities of the post towards those friends on a group level. There are two exceptions in which the user does not directly deploy the groups in her visibility decisions. *First*, certain posts are too privacy sensitive, i.e. it has become a complete regret, or not sensitive at all, in both cases, a binary decision becomes unary, and all user's friends are considered as one group. *Second*, when the number of friends who are or are not supposed to see a post (e.g. one, two or three) is significantly smaller than the number of friend groups shown to the user, then checking the groups requires more effort than just doing a standard search, e.g. typing friend names in a search box. Thus it is no longer necessary to use groups. However, we shouldn't completely disregard friend grouping in such situation, because it can raise the user's awareness about her friends, which can be useful for other aspects of life or later visibility-decision-making. Moreover, if shown appropriately, the friend grouping can help the user spot "unexpected" or "surprising" friends, which then becomes useful for the user to make visibility decisions.

3.2 Why We Use Hierarchical Grouping

In our user study (Section 3.5), we compare the two ways of detecting communities for a Facebook user – Facebook Smart Lists (FSL) and hierarchical modularity-based communities – in terms of their usefulness in facilitating the user's visibility decisions for posting. The original modularity-based community detection algorithm (MOD) takes the user's friend graph as input and produces non-overlapping, flat communities. There is a subgraph corresponding to each detected community of nodes. MOD is then applied to each subgraph, deriving sub-communities. We adapt MOD into a hierarchical one, abbreviated as HMOD. We choose to use HMOD for three reasons: (1) Modularity-based methods are known to have a "resolution limit" problem [22]. It is most likely that, for a community with $\sqrt{m}$ (m is the total number of edges) or less nodes, its sub-communities cannot be discovered. This implies that modularity optimization can miss the substructures of a network. (2) It is well known that people organize semantic concepts hierarchically in memory [13]. The reason for this is because storing generalized information with superset nodes is more economical for humans. Hierarchy is necessary in the navigation for the retrieval of more detailed information. (3) Another incentive that we use HMOD is based on the aforementioned Categorical Thinking, as iterative categorization (i.e. grouping) may be required from the user to make sense of the her friends if the number of friends is simply very large.

3.3 Related Work on Visualizations for Hierarchies

To examine the difference between two friend grouping strategies, there needs to be one common User Interface (UI). Given that a Facebook user usually has hundreds of friends, naively using "pen and paper" to elicit the visibility decisions from the participants may weary them. Bearing this in mind, we decide to let the participants operate on a computer-based UI. For users making visibility decisions with such an interface, we need two basic functions: *First*, browsing is applicable at both group and individual levels. *Second*, making a decision is applicable at both group and individual levels. Various existing works have paved the way for visualizing hierarchical grouping structure. We do not intend to provide a comprehensive review in this subsection. Instead, we give a qualitative treatment to four representative types of visualizations and motivate our design choices. We refer to the two dimensional area on the computer screen where a visualization is rendered as the canvas.

- **Node-Link Diagram** The traditional Node-link Diagrams use shapes (rectangles, circles, etc.) to represent nodes and lines to represent links. The direction from the root of the tree to the leaves is either vertical or horizontal. The nodes (intermediate or the leaves) at the same level need to be aligned at the same vertical or horizontal line. Hence only one-dimensional space is utilized to visualize each level. As shown in Figure 2a, this space can be easily exhausted, especially

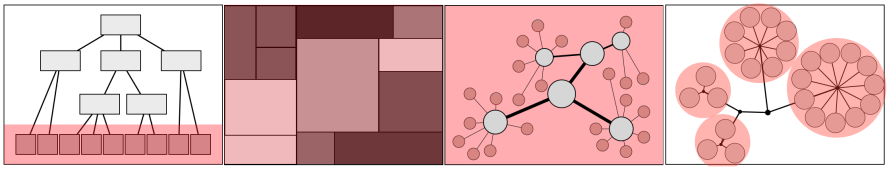

Fig. 2 Four types of representations for visualizing a hierarchical grouping structure, the potential area that can be used to draw leaf nodes is overlaid with red color.

at the leaf level. When the leaves are squeezed to be aligned and fit into the canvas, they easily become too small for the user to interact with, and the grouping structure is no longer clear at the leaf level. Improvements have been made using coloring and merging to reduce the number of branches and/or leaves to draw (e.g. Colored trees [50]). However, they leverage the continuous values of leaves, so that the colors correspond to different average values, giving a sense of numerical ordering. In our case, either the friends or the groups are discrete, which the user needs to differentiate to make a visibility decision. We also note that using the color visual channel to differentiate discrete variables (e.g. Stacked Tree [9]) is problematic, as there are very limited choices for visually distinct colors [31, 28].

- **TreeMap** Grid-based (or matrix-based) visualizations utilize the canvas space more efficiently, as shown in Figure 2b. A typical grid-based layout is treemap [33]. It visualizes hierarchical data by nested rectangles. Many techniques have been proposed to make treemaps more structurally perceivable by humans. For example, shaded colors can bring a sense of ordering to the treemap nodes [54], gradient colors can demarcate different clusters in a treemap (cushion treemap) [58], the aspect ratio of the nodes can be adjusted to improve their readability (squarified treemap) [11]. Compared with node-link diagrams, treemap is more readable for various large-graph-related tasks, but path finding is consistently in favor of node-link diagrams [27]. More importantly, the user cannot conveniently select all the friends in a (sub-)group at once to make a visibility decision in treemaps.
- **Space-Filling Tree** Given the limitations in node-link diagrams and treemaps, hybrid visualizations have been proposed. The space-filling tree [47] is a typical example, as shown in Figure 2c. It spreads the nodes and leaves across the whole canvas. To give a sense of structure, the sizes of the nodes decrease with ascending levels of the tree, the child nodes are mapped in proximity with their parent. But it is probable that, in order to optimally utilize the unoccupied space, the nodes in one branch protrude into the neighborhood of another branch, resulting in a less structural display.
- **Bubble Tree** To heighten the sense of grouping structure, Bubble Tree [29] further constrains the proximity mapping between child and parent nodes – the child nodes are aligned in a circle around their parent, as shown in Figure 2d. This sacrifices potential drawing area on the canvas (still more space-filling than the tra-

ditional node-link diagram), but gains the representation of a stronger grouping structure. The user can select a branch of nodes via their parent node. However, it remains difficult to compare the sizes of the branches on the same level.

3.4 The CircleTree Exploratory Visualization

Considering the previously examined visualizations, we realize that showing the complete structure of a tree of friends may be unnecessary, even interferential to the user. As we try to facilitate the user in determining whether a friend (represented by a leaf node) can see her post, drawing too many intermediate nodes on the canvas produces unnecessary "cognitive overhead" [7], because those nodes not only occupy limited canvas space, but also increase the number of objects that the user needs to process in the limited short-term memory. Therefore, we design a new form of interactive visualization that constrains the number of levels shown (namely one or two levels) and let the user's zooming actions reveal more sub-groups or less only when she needs to. It is also similar to the Bubble Tree in the way that child nodes are positioned in a circle around their parent. We call it CircleTree.

It is important to note that in order to make visibility decisions for a post at the very beginning, the user needs to go through all the friends, regardless of the form of presentation, either simple textual list on a paper or complex visualizations. The benefit of a (good) friend grouping follows after the user's initial contact and familiarization with the generated groups. In other words, the user has made the connection between members and their corresponding group. A group is represented by a token, which can be a shape, a descriptive phrase, or the name of a member from this group, etc. This linkage information is stored in the user's long-term memory. The members can be recalled when the user just sees the group token. In such a way, the user bypasses the serial browsing of each individual member, and directly utilizes a group. The main purpose of our visualization design – CircleTree – is to provide visual tokens for a user's friend grouping. It also adds the elements of structure and engagement to an otherwise lengthy, textual reading and decision-making task. Another purpose of the visualization is to facilitate manual friend-grouping construction, as elaborated in Section 5. The CircleTree visualization is detailed as follows:

A node is represented by a circle, a group of nodes is represented by the circular placement of its child nodes around one extra node, which is the parent node that represents the whole circle, as shown in Figure 3a. With the basic visual principles in mind – that humans are very sensitive to the difference of lightness in grey colors [55], we set the background color white, the friend nodes grey, the parent nodes blue. The latter two colors are also semi-transparent to avoid the occlusion effect. We pick orange and magenta as the highlight color for each friend node and parent node respectively. The large differences (from grey) in saturation and (from blue) in hue promote visual contrast [59]. At first sight, it seems sufficient to use just one visual channel to encode grouping, i.e. the circular placement of child nodes in a group.

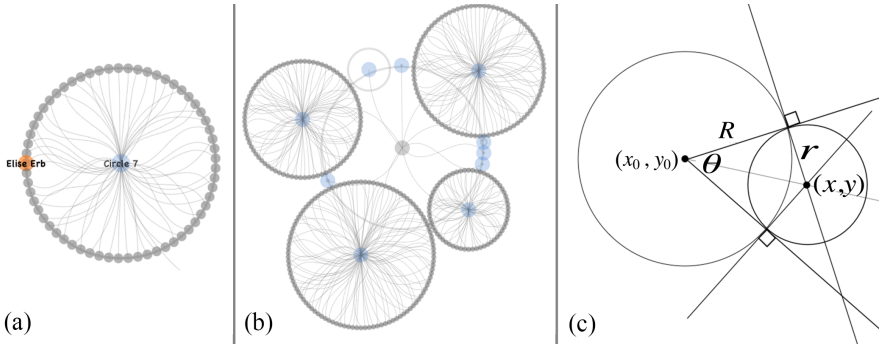

Fig. 3 (a) A single group circle, the grey nodes are the friend nodes, the blue node is the parent of the group circle. (b) The groups are positioned around a central node. (c) An illustration of drawing a group circle around a central node.

But since the user is allowed to drag the nodes to other positions on the canvas, as described below, we add lines connecting the child nodes with the corresponding parent to emphasize that a child node belongs to its parent. The lines within a circle also signal a sense of integration. But in order to avoid overemphasizing the lines instead of the nodes, and sometimes to avoid occlusion between lines and nodes, we choose to increase the transparency of the lines[6]. Furthermore, as argued, curved lines can be used to make certain paths in a graph more apparent [61], based on [20], and curved shapes are often reflective of natural objects, giving the observer a pleasant feeling [34], we choose to use Bézier curves instead of straight lines. However, the exact role that curves play in improving the perception of grouping structure and the aesthetics of the visualization is unclear, and beyond the scope of this work.

The groups are then positioned approximately in a circle around the root node that is under focus, as shown in Figure 3b. In the initial layout, this top node is the root of the tree. We see that the circumference of each group circle formed by its child nodes is naturally scaled with the number of children, presenting a visual order. Every pair of adjacent group circles are tangent to each other. The CircleTree layout algorithm is detailed in Algorithm 1. The radius r_i of an individual friend node from a group circle c is then approximated by $r_i \approx \pi \cdot r/|c|$, where $|c|$ is the number of friends in c, r is the radius of c. Note that very large or small m results in an exceptionally small or large r_i. Thus, minimum and maximum radii r_{min} and r_{max} are set to prevent each friend node from being too small to see or too large that it disturbs the visual ordering. When c has few friends, its assigned r becomes small, making $r_i < r_{min}$. After restoring the overly small r_i to r_{min}, we will likely have relatively large child nodes occupying the entire inner space of c and overlapping with the central parent, which does not make sense to show. Therefore such friend nodes are automatically hidden from sight, instead, the user will only see the grey

[6] Note that this intended reduction of opacity does not make the lines difficult to see on a computer screen, but may lead to sub-optimal printing quality.

Algorithm 1 The algorithm for computing the layout of the group circles around a center (x_0, y_0). Note that (x_0, y_0) can be the position of the root or any center of a parent node of a group circle. We also set the maximum angle for each circle to $\pi/2$, which is an empirically derived value to keep the sizes of the generated circles contained within the canvas. For symbols θ, x_0, y_0, x, y, r and R, please refer to the illustration in Figure 3c.

Require: the array *Arr* storing the sizes of the circles.
n = No.Circles, N = No.Friends, $MaxAngle = \pi/2$.
Let the array *Angles* store the angles θ the circles.
for $i = 0$ to $n-1$ **do**
    $Angles[i] = 2\pi \cdot (Arr[i]/N)$
    if $Angles[i] > MaxAngle$ **then**
        $Angles[i] = MaxAngle$
    end if
end for
Let the array *CS* store the tuples (x, y, r).
if n > 1 **then**
    $x = x_0 + |\tan(Angles[0]/2) \cdot R|$
    $y = y_0 - R$, $r = x - x_0$
    $CS[0] = (x, y, r)$
    $totalAngle = Angles[0]$
    for $i = 1$ to $n-1$ **do**
        $r = |\tan(Angles[i]/2) \cdot R|$
        $s = \sqrt{(r^2 + R^2)}$
        $x = x_0 + \sin(totalAngle + Angles[i]/2) \cdot s$
        $y = y_0 - \cos(totalAngle + Angles[i]/2) \cdot s$
        $CS[i] = (x, y, r)$
        $totalAngle = totalAngle + Angles[i]$
    end for
else
    $CS[0] = (x_0, y_0, R)$
end if
return *CS*

circular silhouette around the parent to mark the visual area of the group, keeping the visualization clean and ordered, as illustrated in Figure 3b.

In the visualization, initially, the user only sees one layer of the tree, as an overview, but can further explore it by the zooming, panning and enabling text labels. We assume that a user can recall her impression of or her relationship with a friend if she see that friend's name. Therefore, when the mouse hovers over a node, the node is highlighted and corresponding label is shown, either a friend name or the name of a numbered intermediate node (e.g. "Circle 5" or "Circle 5.3"). Right-clicking on a parent node maps its child nodes (which we call "the focused children") in a grid layout with the names brought into sight. When a grid layout is triggered, we increase the transparency of all the other nodes on the canvas, so as to reduce the interference from irrelevant visual objects, but still keep them visible in the background to maintain a global context, as shown in Figure 4a. Clicking (left or right) anywhere other than "the focused children" or another parent node

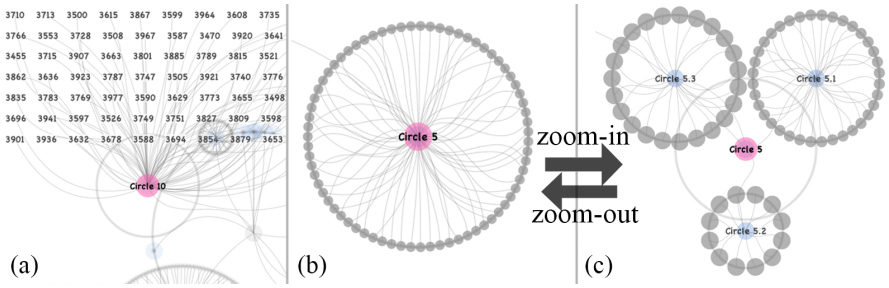

Fig. 4 (a) Right-clicking a parent node reveals the names of friends in that group. (b) A group of friends before zooming-in. (c) The same group of friends from (b) who are further grouped after zooming-in.

on the canvas will restore the original layout. Right-clicking on another parent node will automatically restore the circular placement of the currently focused children, meanwhile shift focus onto the children of the newly clicked parent node. The user can pan (drag to displace visual objects) to adjust the point of interest. If the starting point of panning is not over a node, the whole tree will be panned. If it is over a node, that node will be panned, along with its child nodes if it is a parent.

We take the current mouse position on the canvas as the "anchor point" for zooming actions. An anchor point $P_{anchor} = (x_a, y_a)$ is the position that is invariant during zooming. A zooming action triggers the following transformation: $rt \cdot \beta \cdot (P' - P_{anchor}) = (P' - P)$, in which $P = (x, y)$ is the position before zooming, $P' = (x', y')$ is the position after zooming, $rt \in \mathbb{R}$ is the value of mouse-wheel rotation provided by the operating system, $\beta \in \mathbb{R}$ is a constant adjusting the zooming speed. Note that the zooming speed on X- and Y-axes are the same. It then follows that the scaling factor is $sf = (x' - x_a)/(x - x_a) = (1 - rt \cdot \beta)^{-1}$. During zooming, the radius r_i of each node is multiplied by sf but further constrained by $r_i \in [r_{min}, r_{max}]$. When the child nodes no longer overlap with the corresponding parents, the hidden child nodes and their names are brought into display with zooming-in. When the user zooms into one circle of friends, we perform a "focus-check" to determine whether to further divide the circle. The "focus-check" assumes a rectangular area, half the width and height of the canvas, with the current mouse position as the center point. Upon the user's zooming-in, the only remaining group circle whose parent node is within this area is found and divided. The newly generated sub-circles are presented if the subgraph corresponding to the circle is divisible according the algorithm [46]. We choose the size of the "focus-check" area such that the user does not need to zoom too deeply or too shallowly to explore sub-circles. Zooming out of the visualization makes the sub-circles from the previously divided circle squeezed and overlapped, which will trigger them to merge back to the singular circle again. This is depicted in Figure 4b and 4c.

3.5 Participants, Tasks and Results

In this section, we document the tasks of the participants and the findings from the study.

3.5.1 Participants and Tasks

There were 16 participants (three females), 25-45 years old, from eight countries. Among them were Ph.D students, company employees and graduate students. The participants were equally divided into two groups, which we named directly with the corresponding algorithm abbreviations mentioned in Section 3.2 – HMOD and FSL. Both groups used the same visualization interface detailed in Section 3.4, but with different community detection methods, as their names suggested, namely the hierarchical modularity-based algorithm and Facebook smart lists. Because the latter was not a complete grouping, the friends of a participant that were not in any smart list were put together as one other group.

Our assumption in the user study is that users utilize categories of friends (denoted as C_u) to make a binary visibility decision. We denote the communities that HMOD and FSL produce as C_{HMOD} and C_{FSL} respectively. C_{HMOD} is the result of the interactions between a user and HMOD, with the CircleTree visualization interface. C_{FSL} is the set of non-hierarchical circles of friends constructed from the user's Facebook smart lists, with one extra circle containing the friends who are not in any of the smart lists. Our hypothesis is that, for users' visibility decision-making, C_{HMOD} coincide with C_u, more than C_{FSL}.

We asked the participants to perform the following two tasks: elicitation of regrets in posts and visibility decision-making. In the first task, each participant was asked to identify her regretted posts. In the second task, the participants in the two groups HMOD and FSL were asked to make visibility decisions for each of their posts. Each group has 8 participants and 24 posts. As illustrated in Figure 5, when a participant thinks a friend can see the post, she clicks on the corresponding friend node, the color of which changes to indicate that the post is now visible to the clicked friend. Clicking on the parent node of a group circle toggles every child node's color, or further descendants if some child nodes are already divided by a zooming-in action. The participants could work at their own pace until they were satisfied with their decisions.

Though recent studies have investigated regrets in OSN from different aspects [41, 60], we chose to let the participants explicate their own regrets, as it is easier for a person to make visibility decisions based on her own experience. We collected the posts in face-to-face interviews with the participants. We emphasized the difference between complete and partial regrets. A complete regret meant that the post was supposed to be seen by no one. A partial regret meant that the participant did not mind her post being seen or intended her posts to be seen by some of the friends, but failed to block the other undesired friends. Since a complete regret entails concealing the corresponding post completely, which would render a visibility decision

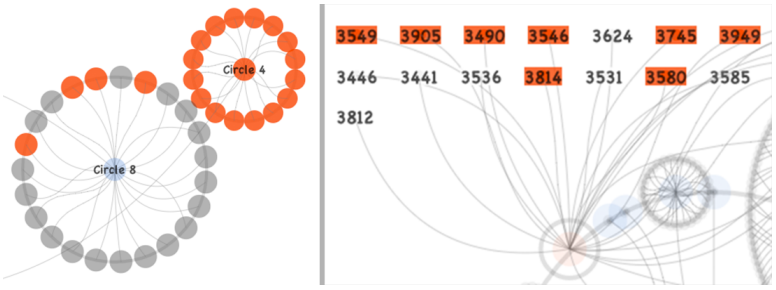

Fig. 5 Participants can determine a post's visibility to each friend individually by clicking friend nodes or collectively by clicking parent nodes in the centers of group circles.

trivial, we guided the participants to only think of partial regrets. Each participant was encouraged to think of at least three posts. A post needs to be specific enough to let the participant define its visibility towards each friend. In total, 48 posts were collected; each participant contributed three personal posts on average. We found that photo-related posts were mentioned frequently, thus making a distinction between photos and topics. Topic-related posts include status updates, web-link sharing and comments.

We recorded the participants' regretted posts and manually classified them into five categories, as summarized in Table 1. The *first* category covers the posted photos that cause embarrassment or awkwardness, typical examples are "drunk party" photos. There are also the photos showing the participant together with some particular person(s), e.g. ex-boy/girl-friend, that the participant feels the need to hide the photos from some friends. The *second* category covers the photos that are less sensitive in terms of embarrassment or awkwardness, but still in need of visibility control. For example, some photos may be so intimate that the participant only wants to show them to her family and best friends. Some photos were taken at a event with a specific group of people, only to whom, as participants argues, the photos should be made visible. More than a third of the posts are photo-related. The *third* category covers the topic-related posts that involve explicit self-expression, including strong opinions and emotional expressions, such as venting negative emotions. Of the seven posts in this category, six are about venting or expressing negative opinions, which the participants felt should be avoided in future, for those posts may harm one's image if disclosed carelessly. The *fourth* category covers the sensitive topics that are less self-involved, but more about the intrinsic sensitive nature of the content of the posts, including politics, religion, sex, race and/or nasty jokes. It is interesting to see that nine out of the twelve posts in this category are about inappropriate jokes. For example, several participants reported that they posted something they believed sarcastically humorous, but in hindsight, they thought it was not wise to expose those posts publicly, as some friends may not understand the humour, or even be offended by it. The *fifth* category covers the relatively less sensitive topic-related posts, which nonetheless need visibility control. For instance, it may not

Table 1 Participants' Regretted Posts

	Categories of Regretted Posts	Frequency
(1)	sensitive photos causing embarrassment or awkwardness	8
(2)	other photos for a specific group of friends	9
(3)	sensitive topics involving emotional expressions	7
(4)	sensitive topics involving nasty jokes	12
(5)	other topics for various specific situations	12

make sense to show the posts to the friends who do not speak the language in which the posts are written.

3.5.2 User Study Results

We use binary entropy to evaluate the effectiveness of the two approaches for users making visibility decisions. $Entropy(post) \in [0,1]$ (Equation 1) calculates the information content (in bits) needed to determine whether a member in a circle can see a post. C is a set of circles of friends, and $c \in C$ generated by HMOD or FSL. $V_{c,post}$ is the number of the friends to whom $post$ is visible in the circle c. N is the total number of friends (including duplicates if circles overlap) in all the circles. $Entropy(post) = 1$ means that on average, in one circle, half the circle can see the post while the other half cannot. This indicates that the given set of circles is unhelpful for the user to make visibility decisions on a group-level, by taking the circles holistically into account. $Entropy(post) = 0$ means that for each circle, the friends in the same circle have the same visibility access to the given post. That is, every circle can be fully utilized by the user to make visibility decisions. The CircleTree visualization in the group HMOD is analogous to a binary-classification tree. Users try to use this tree to make visibility decisions. A"pure" circle in terms of visibility decisions is helpful, since such a circle can be considered as a whole. The initial circles are divided until they are indivisible according to the graph modularity or they are pure. Then the sub-circles are used to calculate entropy scores.

$$\begin{aligned} &Entropy(post) = \sum_{c \in C} \frac{|c|}{N} Entropy(c, post) \text{ with} \\ &Entropy(c, post) = -\frac{V_{c,post}}{|c|} \cdot log_2 \frac{V_{c,post}}{|c|} - \frac{|c| - V_{c,post}}{|c|} \cdot log_2 \frac{|c| - V_{c,post}}{|c|} \end{aligned} \tag{1}$$

Another aspect of a set of visibility decisions for a user's post is its imbalance. That is, the number of friends who can see the post is significantly different than those who cannot see the post. Let V_{post} be the total number of friends who can see the post and $\alpha = min(V_{post}, N - V_{post})$. When α is rather small, e.g. one or two, $Entropy(p)$ can be low almost regardless of which grouping method is used. In such

Table 2 Entropy scores for group FSL and group HMOD, with $\alpha > 1$ and $\alpha > 5$.

	FSL	HMOD
$\alpha > 1$ (24 posts)	**0.46**	**0.20**
$\alpha > 5$ (19 posts)	0.56	0.22

case, while a grouping may still be useful for the participants to browse friends, but it is likely to be less effective for making visibility decisions than the participants just typing individual friend names to search for them in real-time, as discussed in Section 3.1. We know that the average number of friends of each participant is 194. All the 48 posts (24 posts for each group) have $\alpha > 1$ and $\overline{\alpha} \approx 34$. Within these posts, there are 38 posts (19 posts for each group) with $\alpha > 5$ and $\overline{\alpha} \approx 42$. Table 2 shows the average Entropy scores in group HMOD and FSL for $\alpha > 1$ and $\alpha > 5$. Group HMOD achieves lower entropy than group FSL in both cases. This suggests that the circles generated by the hierarchical modularity-based method are taken more holistically into consideration than Facebook smart lists by the participants to make visibility decisions. In other words, it is more often that a circle in the HMOD group, than that in the FSL group, is marked unanimously as the people who "can see" or "cannot see" a post. We can also see that raising α level indeed increases the average entropy scores in both groups, but the increase is more apparent in group FSL ($\approx 22\%$) than in group HMOD ($\approx 10\%$).

We test the statistical significances of the differences between the entropies from HMOD and FSL. It is however, less straightforward to compare the two, because the entropy scores yielded in group FSL are from a different set of participants, with a different set of EOSN and posts. Nevertheless, it is possible to perform an approximate comparison by a pessimistic pair-wise matching. We first calculate the pair-wise squared entropy differences between FSL and HMOD/MOD, deriving a cost matrix, with which, we match the entropies in the two groups via the linear assignment [43] to minimize the sum of the pair-wise differences (so as to minimize the difference between the two models). Based on the resulting pair-wise matches, we perform the t-tests. It then follows that, in comparing HMOD and FSL, the t-statistic is 9.146 for $\alpha > 1$ and 12.810 for $\alpha > 5$. The t-statistics reject the corresponding null hypotheses with two-tail Confidence Interval (CI) $= 99.9\%$ and one-tail $CI = 99.95\%$. It is then evident that HMOD is significantly better than FSL.

From this user study, we gain more insight into users' privacy decision-making process in online posting. *First*, it is evident that categorical thinking is used when users make binary visibility decisions. *Second*, graph-modularity-based friend communities assist users more efficiently for such decisions than the profile-attribute-based Facebook smart lists. This implies that the former produces the communities that fit the categories of friends that a user has in mind, more than the latter. We examine another state-of-the-art community detection algorithm for EOSN in comparison with the modularity-based algorithm in Section 4 and discuss the implications of the results. Third, the essence of categorical thinking is to reduce the cognitive load. If there are too many information objects (in our case, online friends), hierar-

chical categorization supports the users' visibility-decision-making. When the resolution limit of MOD is reached, the sub-circles can be of more help for the users. The results also give us guidance in designing information visualization systems – categorization and abstraction are important for users to process large amount of information.

4 Community Detection and Social Groups

In this section, we first introduce the two community detection methods of interest (Section 4.1), then describe the datasets (Section 4.2) on which the two algorithms run. We compare and discuss the performances of the two algorithms on these datasets (Section 4.3), and propose an extension of one of the algorithms to accommodate the overlapping nature of online friend groups (Section 4.4). We summarize and compare the performances of all three algorithms by the end of this section.

4.1 Two Models for Community Discovery

From our preliminary user study, we know that, in order to make sense of the friends in one's online social life, it is important to categorize them, either for the ease of processing and memorizing friends' information, or as an efficient means for making decisions. Given the large number of friends that one usually has in EOSN, automated community detection can be very helpful not only as the basis for visibility decisions, as investigated in the previous section, but also for other tasks in online contact management, such as simply keeping an overview, sorting incoming messages, etc. In this section, we examine community detection algorithms and their relationship with real-life social groups. We compare two models for community detection in EOSN: the graph-modularity-based model (MOD) using eigenvalue decomposition [45] and the Generative Model for Friendships (GMF) [39].

Modularity is the number of edges falling within groups minus the expected number in an equivalent network with edges placed at random [46]. Larger modularity value suggests more obvious community structure in the graph. There exists abundant and different techniques that optimize the modularity of a graph. We chose to implement Newman's spectral optimization algorithm that iteratively bisects a given graph using the eigenvectors of the modularity matrix. This approach is generally more accurate than the techniques such as greedy methods and external optimization, and less computationally expensive than global optimization approaches such as simulated annealing [21]. We also implement vertex-moving to improve the final modularity score, as proposed in [46]. Intuitively, in each bisection of the input (sub-)graph, "vertex-moving" moves one vertex at a time, from one (sub-)community to the other, if the modularity is increased, it makes this move permanent. The average, combined complexity of this algorithm is $O(N^2 log\ N)$.

GMF is a recently proposed community detection model that leverages both the friend-profile features and the friend-graph structure in an EOSN [39]. The resulting communities have the following properties: (1) the friends in the same communities have common features, such as education, work; (2) different communities may emphasize different features; (3) the communities may overlap. GMF has been evaluated against the ground-truth communities from three EOSN datasets (as described in Section 4.2), and compared with eight baseline models – Mixed Membership Stochastic Block Models, Block-LDA, K-means clustering, Hierarchical Clustering, Link Clustering, Clique Percolation, Low-Rank Embedding and Multi-Assignment Clustering (as elaborated in [39]). It was demonstrated that GMF generated more accurate communities than the baselines.

4.2 Three EOSN Datasets

The three datasets were collected from Facebook, Twitter and Google+, which are available online[7]. We downloaded these datasets, removed empty files, and discarded the ego-networks whose ground-truth circle(s) contains just one friend. Finally we obtained 10, 909 and 129 ego-networks from Facebook, Twitter and Google+ respectively, which we use for our experiments. Note the data is a subset of the data used in [39]. Each ego-network includes the user's and the friends' profiles, the friend graph and the set of manually constructed circles by the user. For the Twitter and Google+ friend graphs, we ignore their directivity as MOD runs on undirected graphs. We denote the complete set of friends as V, the friend nodes retrieved from the user's ground-truth circles in an EOSN as $V_{circles}$, the friend nodes retrieved from the user's friend graph as V_{edges}, a ground truth circle as c, the set of ground-truth circles as C, an algorithm-generated circle as c' and a set of algorithm-generated circles as C'. The three datasets are summarized in Table 3. We see that $\overline{|V_{circles}|} < \overline{|V_{edges}|}$ for the three datasets, since $V_{edges} \subseteq V$, it indicates that $V_{circles} \subset V$. Moreover, we observe that overlapping ground-truth circles are common, but also limited such that a friend is usually assigned to less than two circles.

4.3 Performances of GMF and MOD

We follow the same method and metrics in [39] to evaluate how well a set of generated circles C' match the user's manual circles C. Balanced Error Rate (BER) [12] and F1 scores are used to measure the matches of circles, as defined in Equation 2 and 3. We use $RBER(c,c')$ to refer to $1 - BER(c,c')$. In order to determine which $c' \in C'$ corresponds to which $c \in C$, we perform a linear assignment using the Hungarian Algorithm [43] to maximize the sum of the pair-wise $RBER$ or $F1$.

[7] https://snap.stanford.edu/data/index.html#socnets [Accessed on Dec 9, 2013]

Table 3 Three ego-network datasets summarized, from left to right: $\overline{|V_{circles}|}$ is the average number of friends from a user's ground-truth circles, $\overline{|V_{edges}|}$ is the average number of friends from a user's friend graph, $\overline{|C|}$ is the average number of a user's ground-truth circles, $\overline{|c|}$ is the average ground-truth circle-size, $\overline{No.Comms.P}$ is the average number of ground-truth circles to which a friend belongs.

EOSN	$\overline{\lvert V_{circles}\rvert}$	$\overline{\lvert V_{edges}\rvert}$	$\overline{\lvert C\rvert}$	$\overline{\lvert c\rvert}$	$\overline{No.Comms.P}$
Facebook (10)	298	423	19.3	26	1.6
Twitter (909)	36	134	4.4	12	1.4
Google+ (129)	304	1948	3.6	135	1.6

$$BER(c,c') = \frac{1}{2}\left(\frac{|c \backslash c'|}{|c|} + \frac{|c' \backslash c|}{|V_{circles}| - |c|}\right) \tag{2}$$

$$F1(c,c') = 2\frac{|c \cap c'|}{|c| + |c'|} \tag{3}$$

We ran GMF[8] and MOD on the ego-networks that only included the friend nodes from ground-truth circles, so that we could compare C and C'. The reason that we ran GMF again instead of directly using its original result was because of the incomplete ego-network data that we could download and some trivial data (e.g. an ego-network containing only one friend) that we discarded afterwards. As such, both GMF and MOD were run on the subsets of the ego-networks that were described in [39], namely 10 Facebook, 909 Twitter and 129 Google+ ego-networks instead of 10, 1000 and 133 ego-networks. Due to the complexity of the algorithm (with the worst case complexity $O(N^3)$, N being the number of friend nodes in an ego-network), we ran GMF for each ego-network with selective K values (the number of communities), $K = 3, 5, 7$ and 9 respectively. Then we select the K value that corresponds to the highest average $\overline{RBER}$ or $\overline{F1}$, and match C and C' for each ego-network via linear assignment. As for MOD, K is automatically derived in the process of modularity maximization. The results are summarized in Table 4 and 5. Note that while certain K of GMF achieves the highest $\overline{RBER}$, it does not necessarily mean this K corresponds to the highest $\overline{F1}$. Thus we have two different sets of combinations of Ks with respect to the $\overline{RBER}$ and $\overline{F1}$ measures. The columns $\overline{|C'|}$ and $\overline{No.Comms.P}$ in Table 5 are based on the average values of these two sets of Ks.

We denote the GMF algorithm that was run on the original ego-network datasets, with a full range of K values checked, as GMF0. This is to differentiate it from the GMF model that we ran on the subsets, with the four K values checked. From Table 4, we notice that the *RBER* and $F1$ scores of GMF on the Facebook and Google+ datasets are smaller than those of GMF0, and the *RBER* and $F1$ scores of GMF on the Twitter dataset are comparable to or higher than those of GMF0. The relatively

[8] The code can be downloaded from the author's web page: http://i.stanford.edu/ julian/. We used the default parameters in the code with different K values.

Table 4 The comparison between the results of GMF running on the subsets with four K choices (white columns) and the original sets (gray columns) of the ego-networks: Facebook (Fb), Twitter (Tw) and Google+ (Gp).

GMF	Fb(10)	Fb(10)	Tw(909)	Tw(1000)	Gp(129)	Gp(133)
$\overline{RBER}$	0.83	**0.84**	**0.77**	0.70	0.65	**0.72**
$\overline{F1}$	0.53	**0.59**	0.32	**0.34**	0.24	**0.38**

Table 5 The results of running GMF and MOD on the three subsets of ego-networks. The gray sub-columns are the results for GMF, the white ones are for MOD. $\overline{|C'|}$ is the average number of generated circles, $\overline{|c'|}$ is the average size of each generated circle, $\overline{No.Comms.P}$ is the average number of circles to which each friend belongs.

EOSN	$\overline{RBER}$		$\overline{F1}$		$\overline{\lvert C'\rvert}$		$\overline{\lvert c'\rvert}$		$\overline{No.Comms.P}$	
Facebook	0.83	**0.86**	0.53	**0.67**	3.3	7.0	90	41	1.5	1
Twitter	0.77	**0.81**	0.32	**0.68**	5.2	3.0	7	12	2.7	1
Google+	0.65	**0.75**	0.24	**0.62**	6.9	3.1	44	98	3.8	1

large performance difference on Google+ is due to the limited choices of K in GMF. From Table 5, we see that MOD fully outperforms GMF on $RBER$ and $F1$ measures.

4.4 Multi-membership Modularity-Based Method

From Table 5, we can also see that MOD generates the $\overline{|C'|}$ that is closer to the ground-truth as shown in Table 3. We also know that though overlapping circles are common in the ground-truth, one friend is rarely put into more than two circles, whereas GMF on Twitter and Google+ generates the circles that have $\overline{No.Comms.P}$ equal to or larger than three, which led to its relatively low performance on these datasets. However, a significant limitation of MOD is that it produces non-overlapping communities, while it is obvious that OSN users construct overlapping circles by themselves. Thereby, we propose an extension of MOD that allows multiple circle memberships, which we call Multi-membership Modularity-based community detection, shortly as MMOD. We define a metric we call the External Belongingness (EB as in Equation 4), in which $neighbors(v,c')$ is the number of neighbors (one hop away on the friend graph) of a given friend v in an external circle c', $degree(v)$ is the degree of v. c' is external to v if $v \notin c'$. We first run MOD to derive a set of non-overlapping circles. Then for each friend, we obtain a list of external circles (the circles to which the friend does not belong) with the corresponding EB scores. We subsequently check the highest EB score for each friend, if it exceeds the previously defined θ_{EB}, the friend is assigned to the corresponding external circle. In this way, we obtain a set of overlapping circles with some friends belonging to two circles. However, it remains the question of how to select θ_{EB}. We

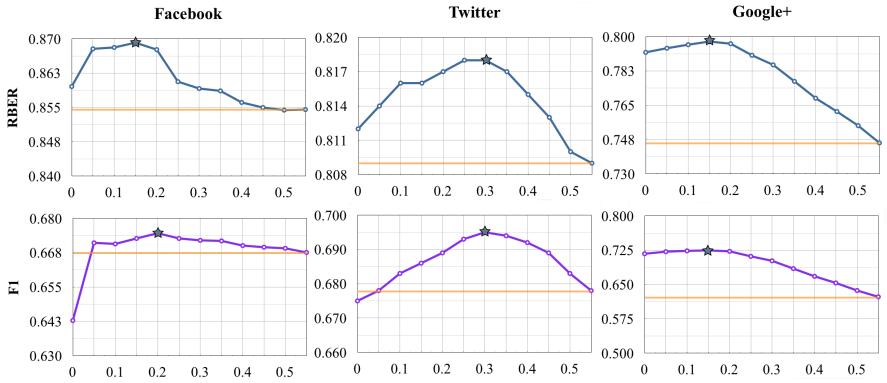

Fig. 6 The *RBER* and $F1$ performances of MMOD with different θ_{EB} values. The baselines are drawn to indicate the corresponding MOD performances and the stars are to mark the optimal θ_{EB} points.

run MMOD with different $\theta_{EB} \in [0, 0.5]$ with the step size 0.05. Then we match the respective overlapped C' with C, the performances are plotted in Figure 6. In each plot of Figure 6, the last point is the average *RBER* or F1 score from MOD, through which a straight horizontal line is drawn to indicate baseline performance. The point with the highest performance is marked with a star.

$$EB(v, c') = \frac{neighbors(v, c')}{degree(v)}, v \in V, v \notin c' \tag{4}$$

From Figure 6, we can see that the performances of MMOD are generally better than those of MOD. The curves also follow the similar trend that increases till some particular θ_{EB} and drops. Around $\theta_{EB} = 0.5$, rarely any friend nodes can be found in external circles, thus the performances regress to be close to MOD's. We also find that the optimal threshold θ_{opt} values for Facebook and Google+ data are similar, which stay around 0.15 for both *RBER* and $F1$, whereas for Twitter data, this value is 0.35. The *RBER* and $F1$ scores of MMOD at these θ_{opt}, along with other results (the same columns as Table 5) are summarized in Table 6, from which we see that MMOD fully outperforms MOD, and that the $\overline{No.Comms.P}$ values are very close to those of the ground-truth datasets. The better results on MMOD also have the implication that people indeed tend to put the friends who are the connectors or hubs in the ego-network into different circles at the same time. We summarize the performances of GMF, MOD and MMOD in Figure 7.

We also observe that θ_{opt} empirically correlates with the average size $\overline{|V_{circles}|}$ of an ego-network, which is around 300 on Facebook and Google+, and 30 on Twitter. For instance, we can describe this relation with Equation 5. If we consider the MMOD-generated circles match the user's manual circles better (indeed, the *RBER* rates are close to or well above 0.8), the relation in Equation 5 suggests that, on the one hand, users tend to manually create less overlapped circles when they have fewer friends. On the other hand, θ_{opt} decreases exponentially slower than the num-

Table 6 The results of running MMOD on the three subsets of ego-networks. $\overline{|C'|}$ is the average number of generated circles, $|c'|$ is the average size of each generated circle, $\overline{No.Comms.P}$ is the average number of circles to which each friend belongs.

EOSN	$\overline{RBER}$	$\overline{F1}$	$\overline{\lvert C'\rvert}$	$\overline{\lvert c'\rvert}$	$\overline{No.Comms.P}$
Facebook	0.87	0.67	7.0	52	1.3
Twitter	0.82	0.70	3.0	18	1.5
Google+	0.80	0.73	3.1	166	1.7

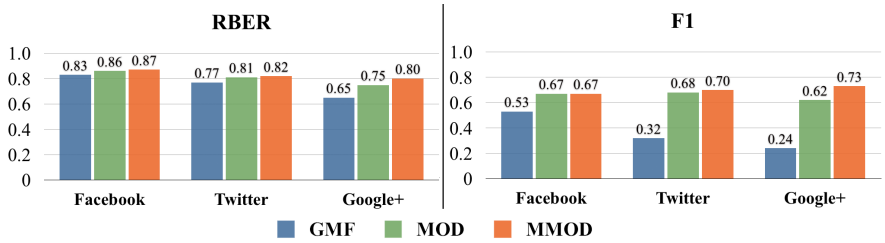

Fig. 7 The overview of the performances of GMF, MOD and MMOD.

ber of one's friends increases, which means that on a relatively large scale (e.g. $|V_{circles}| \in [100, 1000]$), given that EOSN are often sparse [57, 42], users' θ_{opt} for allowing a friend to be in multiple circles remains similar ($\theta_{opt} \in (0.12, 0.20)$ approximately). However, in order to accurately capture the relationship between the number of friends and the optimal threshold, we need a further investigation. It may involve other potentially correlated parameters, more sophisticated models and more data, which is beyond the scope of this work. Equation 5 is manually derived based on the observations from Table 3 and Figure 6. It serves as an intuitive guidance for determining θ_{opt}.

$$|V_{circles}| = 3 \times 10^{(\frac{0.3}{\theta_{opt}})} \Longleftrightarrow \theta_{opt} = \frac{0.3}{lg|V_{circles}| - lg3},\ \theta_{opt} > 0 \tag{5}$$

We perform ANOVA (ANalysis Of VAriance) to compare GMF, MOD and MMOD on the three datasets. The **p** values are summarized in Table 7. We can see that the **p** values on the Facebook dataset are rather high, and the **p** values on the other two datasets are low ($\mathbf{p} < .001$). This means that the variance between the three models is not significant on the Facebook dataset, but very significant on the Twitter and Google+ datasets (in fact, the F-statistics on these two datasets approach the ends of the corresponding F-distribution curves.) In Figure 7, the observed differences were statistically significant for both $\overline{RBER}$ and $\overline{F1}$ on the Twitter and Google+ datasets (all $\mathbf{p} < .001$ for one-way ANOVAs), but not for the Facebook dataset ($\mathbf{p} = .43$ for $\overline{RBER}$ and $\mathbf{p} = .13$ for $\overline{F1}$). The latter may be a result of the small sample.

Table 7 The **p** values of the ANOVA for GMF, MOD and MMOD, of both $\overline{RBER}$ and $\overline{F1}$measures, on the datasets of Facebook, Twitter and Google+ respectively.

	Facebook	Twitter	Google+
$\overline{RBER}$	.43	$< .001$	$< .001$
$\overline{F1}$	.13	$< .001$	$< .001$

4.5 Discrepancy between Predicted and Manual Circles

Though a community discovery algorithm can predict reasonably good circles, it is unlikely that it can make a perfect prediction. This attributes to the fact that manual circle-creation process is inherently subjective, and varies on the same person for different purposes. The ground-truth circles of the ten Facebook users that we used in our experiments were obtained by a Facebook app[9], in which the user entered comma-separated category labels for each friend. Existing labels could be reused by a selection from a drop-down box. Each label represented a circle to which a friend belonged. The text cue for entering the label(s) for each friend **Fr** was "I know **Fr** because ..." followed by the label-entering text-field. In another exercise [15] of friend-grouping, the groups (i.e. circles) were constructed by "card sorting". The name of each friend of a participant's was printed on a paper card. Several cards were randomly selected and spread on a table, the participant was then asked to assign the rest of the cards to the selected ones to form groups. We can see that the Facebook app friend-grouping exercise encourages more overlapping circles to be created than the card-sorting exercise.

Different user interfaces may directly reflect intrinsic and systematic differences on a functional level, rather than on a perceptual level. Facebook provides a social platform mainly for mutual friends – two people become friends when one "accepts" the other's "friend request". The friends of friends are recommended if the user wants to add more friends. Twitter and Google+ implement a "follower-followee" mechanism, which means a friendship is not necessarily reciprocal. On Twitter, the user clicks the "follow" button to follow a "friend", every newly followed friend is not necessarily put into a friend list (i.e. a circle), whereas on Google+, the "follow" button becomes the "add" button, and every newly followed friend has to be added into one of the existing circles or a new one. This is an important reason that the number of friends in Google+ circles is much more than that in Twitter lists, as shown in Table 3. We see that people create circles differently under different circumstances, consciously or unconsciously. It is therefore important to create interfaces that help users gain insights about their EOSN friendships from different aspects, and let them form their own friend circles with more informed decisions.

Moreover, social and cognitive theories shed light on human social grouping behavior and inform computer scientists to design community detection algorithms

[9] https://www.facebook.com/apps/application.php?id=201704403232744 [Accessed on Dec 12, 2013]

and interactive visualizations. The social brain hypothesis (SBH) offers a framework for integrating evolutionary and social psychological perspectives on human social complexity. SBH predicts a natural community size of around 150 for modern humans (Dunbar's number [18]), and now there is considerable evidence confirming that this is the typical size of both personal social networks and key types of human community [17]. Note that 150 is the typical size of a person's active network, in which she knows how these the friends fit into her social world and they know how she fits into theirs [17]. From the literature in cognitive science, we also know that there is the cognitive capacity limit in human Short-Term-Memory (STM), which is inline with the theory of categorical thinking (Section 3.1). This capacity limit is averaging on seven [40], which means that people can remember seven chunks of information in STM tasks. In our case, we can consider a chunk to be a group of friends. This limit is subject to debate, later evidences showed that it was a high estimate, lower numbers were proposed, e.g. four [14]. The theories on social group size and human's cognitive capacity limit provide more incentives for interactive visualizations, which should enable users to flexibly interact with friend visual objects on different granularity-levels – from (sub-)groups of friends to individual friends.

5 Improving the Tool Design

From the previous sections, we understand that grouping friends is important for OSN users to manage online contacts and make privacy decisions. A carefully designed community detection algorithm can produce decent friend circles that match users' manual circles, but this matching is hardly perfect due to the subjective nature of friend grouping. To close the gap between computer-based grouping and human grouping, tools need to be designed and built. The goal of such tool that leverages interactive visualization and accurate community detection is not only to show its users their structured friendships, but more importantly, to make the structures more usable for the users.

We have introduced a tool in our user study to assist users' visibility decision-making (Section 3). It visualizes the generated circles of the user's friends, and allows hierarchical exploration. However, this tool addressed only part of the information about the user's friends, with limited navigation functions. As various taxonomies for visual analytics or information visualization unanimously emphasized [25, 32, 63, 53], presenting multiple aspects and providing multiple perspectives are essential for visualizing large and complex data. We developed a new online application named FreeBu[10]. We motivate and describe three more views that supplement the CircleTree view (Section 3.4). All the four views serve a two-fold purpose: (1) to provide users with different insights about their own ego-networks, (2) users can manually construct their Facebook friend lists with the tool.

[10] http://people.cs.kuleuven.be/ bo.gao/freebu/ [Accessed on Dec 12, 2013]

From Section 3 we have known that FSL are less efficient in visibility decision-making than the communities generated by MOD, suggesting that graph-based communities coincide more with the friend groups that a user has in mind. An investigation (Section 4) in the three community detection algorithms GMF, MOD and MMOD has shown that the ground-truth circles are still in favor of the graph and modularity-based methods. And introducing overlaps further increases the *RBER* and $F1$ accuracies. However, it is incorrect to assume the human friend-grouping process is systematically similar to the algorithmic process just because both produce similar groups. The CircleTree visualization shows circles as friend groups, in which each member is labeled by the name. Other types of data, such as profile, posts, chat history, friend graph, etc. are also potentially useful for the user's understanding of her own ego-network. They can inspire the user to reflect on her online contacts and facilitate friend-group creation.

As the recent study [15] on OSN-friend-grouping shows, people do consider attributes, such as school, music band or youth community, when they group friends, we refer to this type of grouping strategy as the *Attribute* strategy. Also, people indeed tend to put the friends who are mutually friends into the same the group, we refer to this strategy as the *Graph* strategy. Another graph-related, but slightly different grouping strategy is based on some particular friends – "I know those friends via this friend", we refer to it as the *Connection* strategy. The fourth strategy is based on trust or closeness, to which we refer as the *Closeness* strategy. Informed by these grouping strategies, we have four visualizations in FreeBu to accommodate users' comprehension of their online friends and help users create friend groups semi-automatically. The four visualizations/views are described as follows:

- **Circle View** The circle view (i.e. the CircleTree Visualization) is for the *Graph* strategy. Mutually connected friends tend to be put in the same circle, the user can drag and drop a circle or an individual node to compose her own Facebook friend list (Figure 8a). The group circles with different sizes also provide the user with a sense of ordering, helping her quickly find outliers or surprising circles. The visualization and interaction strategies are detailed in Section 3.4.
- **Map View** The map view is for the *Graph* and the *Connection* strategies. The user's friend graph is directly visualized in a force-directed layout with the Fruchterman-Reingold algorithm [23], which is a typical graph-layout algorithm. It pulls connected nodes together and pushes disconnected nodes apart. Users can easily observe visual clusters and hub-nodes. The user can zoom and pan to explore the graph, zooming-in brings out the node labels. Mouse-hover on a friend node also brings out the friend's name label, meanwhile highlights the connections of this friend on the graph. Right-clicking a friend node will automatically select this node as well as its neighbors. User can then drag and drop the selected nodes to compose her own friend list (Figure 8c). Furthermore, the nodes' radii are set proportionally to the corresponding Betweenness [44] scores, so that the important nodes that connect different parts of the user's ego-network are enlarged and emphasized. It has been shown that the bridging structure in a user's EOSN is important for predicting strong social ties, such as romantic partners [4]. To make the group-creation more flexible, the point-in-polygon function is

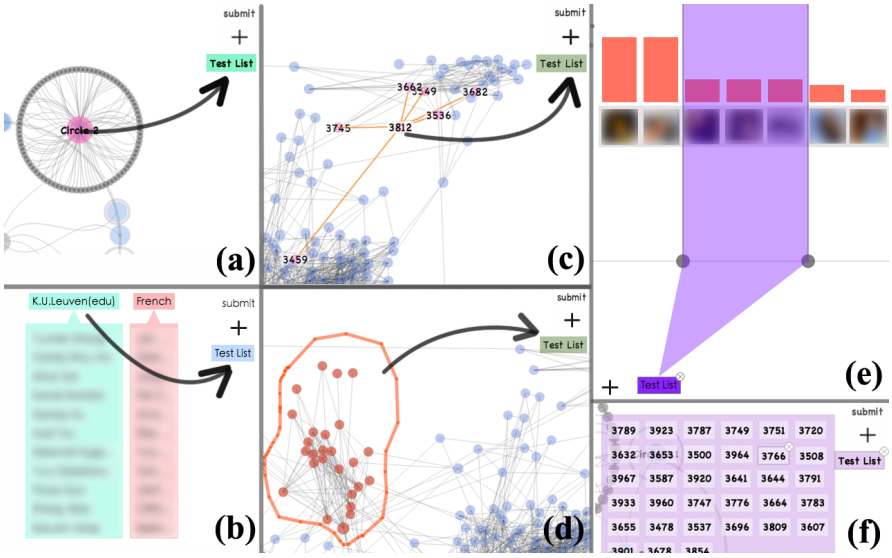

Fig. 8 This figure shows the four views in FreeBu and the drag-drop actions to compose user-defined lists. Each arrow indicates a group-level drag-drop action.

implemented. The user can turn on this function by pressing the "pen" button on the bottom-right corner of the canvas and draw a polygon to enclose and select the nodes of interest, and drag-drop the selected nodes to compose lists (Figure 8d).

- **Column View** The column view is for the *Attribute* strategy. The column view generates the groups of friends based on common profile-attributes between friends, which is a generalization of Facebook smart lists. Each column represents a group. The "head" of the column is labeled with the corresponding attribute-value name. The "body" is a stack of friend name tags belonging to that column. If a column contains more than N_{col} (e.g. $N_{col} = 12$), only N_{col} friend tags are initially shown in the body of the column, with the "..." symbol to indicate there is more tags. Mouse-hover on the head of a column expands the column and show all the member names. The heights of the columns are proportional to corresponding the numbers of friends. Users can scroll left or right with mouse wheel to explore the columns. They can click the "overview" button for a summary of all the column labels. The user can drag and drop a column or an individual tag to compose lists (Figure 8b). Moreover, the user can drag and drop columns into the "intersection" area at the bottom of the canvas. This area keeps the members that satisfy the attribute values from the columns. The user can then use intersected area (also via drag-drop) to compose her friend lists.
- **Rank View** The rank view is for the *Closeness* strategy. Studies [56, 17] have shown that interaction frequency linearly corresponds to the strength of interpersonal ties. We visualize the users' friends by aligning their profile photos horizontally near the middle of the canvas. The photos are ranked according to the

communication frequencies of the user with her friends in Facebook chat. On top of each photo, a bar is shown if there is a communication history of the user with that friend. The more frequently the user chatted with a friend, the higher the bar is. The user can scroll left or right with mouse wheel to see the bars and photos. Mouse-hover can enlarge a photo can brings out the name beneath it. The user can select one or more friends by moving the two "knobs" with vertical lines. Clicking on a user-defined list "absorbs" the friends that are "clipped" by the two knobs into that list (Figure 8e).

The four views share a similar way for creating customized friend lists. The user starts by clicking the "plus" button to add a new, empty list, aligned on the right (in the first three views) or the bottom (in the rank view) of the canvas. Each list is shown as a rectangle. The user can right-click a list to edit its name. Drag-drop actions put selected friends into a list, as illustrated in Figure 8a-d, whereas in the rank view, "clipped" friends are put in a list by user clicking on the list, as shown in Figure 8e. Mouse-hover on a list brings out the friend-name tags of the list in a grid layout. Mouse-hover on a list or a tag also brings out the "remove" button, as shown in Figure 8f. In this way, the user can remove a list or a member if needed. The user can submit the lists to her Facebook account by clicking the "submit" button.

An elaborate multi-method user study on the usefulness and perceived values of FreeBu is beyond the scope of this chapter. We refer to [16] that has detailed such study. Through a factor analysis, it showed that FreeBu received high scores (between 4 and 6 on a 7-point Likert Scale) on several factors of perceived values. These factors include Audience Control and Audience Reflection. The first factor refers to sharing information with differentiated friends. For example, "FreeBu helps me create Facebook friend lists". The second factor refers to the reflection and re-evaluation of one's friends in her EOSN. For example, "FreeBu clarifies my relationships with others of whom I am not fully aware". The regression analyses in [16] further identified several attributes that directed users' attention and guided users' usage of the tool. For example, in the map view of FreeBu, users are more interested in the friends who act like hubs (with high betweenness scores) or the friends who are outliers (with low degree scores) in their social networks. In the rank view, users were very interested in the friends to whom they often communicated.

6 Conclusion

In this section, we first summarize our research, then address the future work.

6.1 Summary

In this work, we addressed the issue related to privacy-decision-making in Online Social Networks (OSN). The available large amount of information about friends

overwhelms a user. It is then difficult for the user to decide the audience for her online posts. Various research work has pointed to friend categorization. Indeed, the theories in categorical thinking and social networking limits provide us with further support. Leveraging humans' innate ability to process visual information, we developed an online visualization application to help users explore and group friends. It requires careful design choices in both visualizations and algorithms. We first reviewed various existing tools, identified their merits and limits. We then described our first tool based on the CircleTree visualization and the modularity-based community detection (MOD). The former is our new visualization design. We conducted a user study to investigate OSN users' visibility-decision performances with two different grouping methods, under the CircleTree visualization. The participants were divided into two groups, one used hierarchical, modularity-based community detection method (HMOD) interactively, the other used Facebook smart lists (FSL). We found that the former group of participants utilized the circles more efficiently the latter. This provides the evidence that HMOD is more supportive than FSL for visibility decisions. It also suggests that graph-based algorithms can produce the communities that match users' manual circles, more than attribute-based ones.

We then compared MOD with another community detection model, Generative Model for Friendships (GMF). It had been shown that GMF outperformed the other eight community detection models [39]. The corresponding nine algorithms were run on three ego-network datasets, and compared to ground-truth circles. We ran MOD and GMF on the sub-datasets (due to the availability of the data), and found that MOD outperformed GMF. We also examined the characteristics of the ground-truth circles and proposed the Multi-membership Modularity-based community detection method (MMOD) that produced overlapping communities, with similar overlapping rate to the ground-truth. We then found that MMOD outperformed MOD.

It is important to note that improving community detection algorithms alone is insufficient. Users need informative visualizations to comprehend her online friends and construct her own friend lists. Guided by relevant sociological research and visualization design taxonomies, we developed three more interactive visualizations that compensated the CircleTree visualization. The four visualizations are based on four different friend-grouping strategies. They incorporate similar list-construction user interfaces.

In summary, this work begins with the concerns for online privacy and contact management, results a web application for EOSN friend-exploration/grouping. We examined in detail the design choices from different perspectives: information visualization, community detection algorithms, human cognition for visual perception and information processing, and social theory on social groups.

6.2 Limitations and Outlook

We identify several main improvements for FreeBu:

- In the four views, each friend is only represented by her name (the rank view also includes photos). More information, such as photo, profile, recent status and likes can be summarized in an "info box" that appears besides each focused friend.
- There often exist the friends who do not connect to other friends in an ego-network. The loners can be randomly mixed into the circles in the circle view or scattered in the force-directed graph layout in the map view. It is then more orderly to collect these loners into the same circle or to map them in proximity in the graph.
- We can improve the circle view by applying MMOD (Section 4.4).
- For the circle view, we notice that the user needs to zoom-in fairly deeply to reveal the friend names in a circle. This can be improved by modifying the label-revelation threshold. Also, the positioning of the name labels needs adjustment, so as to avoid overlaps, while maintaining a grouping structure.
- The graph layout in the map view can be colored according to the communities detected by MOD, similar to InMaps (Section 2). The drawback of discretizing community colors is that it ignores the continuity of the friend graph. Some friends are meant to be community-ambiguous. One way to address this issue is via gradient colors. First, a friend's membership to the circle is characterized some measure, e.g. its clustering coefficient (as that in Social Graph in Section 2). However, we need to be more careful to make people perceive such fusion as a natural transition between communities on the graph.
- Because zooming can create too much local focus and lose global context, it could be more helpful for users to add the "fisheye-view" [24] and "map-window" [7] functions in the circle and map views.
- In all the four views, we can add filtering and searching function to improve users' exploratory experience.
- FreeBu users have reported in some cases rendering visualizations is slow. Complex visualizations and user interactivity occupy a large part of browser resources, sometimes result slow response or crash. Though the current standard web technologies are encouragingly evolving, such as improved graphics rendering capabilities in HTML5, faster built-in Javascript engines, the browser-based computation power is still limited for large-scale, online, interactive visualizations. For tools like FreeBu, visualization programs need to be more economic.

Acknowledgement

We thank the Flemish Agency for Innovation through Science and Technology (IWT) and the Fonds Wetenschappelijk Onderzoek – Vlaanderen (FWO) for support through the projects SPION (grant number 100048) resp. Data Mining for Privacy in Social Networks (grant number G068611N).

References

[1] Acquisti A, Gross R (2006) Imagined communities: Awareness, information sharing, and privacy on the facebook. In: Privacy enhancing technologies, Springer, pp 36–58

[2] Allport GW (1979) The Nature of Human Prejudice. Basic books

[3] Auber D, Mary P (2013) Tulip – an information visualization framework dedicated to the analysis and visualization of relational data. URL (Weblog): http://tuliplabrifr/TulipDrupal/ (Download: 1129 2013)

[4] Backstrom L, Kleinberg J (2014) Romantic partnerships and the dispersion of social ties: A network analysis of relationship status on facebook. In: Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing, ACM, pp 831–841

[5] Bakewell S (2010) How to live: A life of Montaigne in one question and twenty attempts at an answer. Random House

[6] Bastian M, Heymann S, Jacomy M (2009) Gephi: an open source software for exploring and manipulating networks. In: ICWSM

[7] Beard D, Walker J (1990) Navigational techniques to improve the display of large two-dimensional spaces. Behaviour & Information Technology 9(6):451–466

[8] Bernstein MS, Bakshy E, Burke M, Karrer B (2013) Quantifying the invisible audience in social networks. In: ACM SIGCHI Conference on Human Factors in Computing Systems (CHI 2013).

[9] Bisson G, Blanch R (2012) Improving visualization of large hierarchical clustering. In: Information Visualisation (IV), 2012 16th International Conference on, IEEE, pp 220–228

[10] Boyd DM (2008) Taken out of context: American teen sociality in networked publics. ProQuest

[11] Bruls M, Huizing K, Van Wijk JJ (2000) Squarified treemaps. In: Data Visualization 2000, Springer, pp 33–42

[12] Chen YW, Lin CJ (2006) Combining svms with various feature selection strategies. In: Feature Extraction, Springer, pp 315–324

[13] Collins AM, Quillian MR (1969) Retrieval time from semantic memory. Journal of verbal learning and verbal behavior 8(2):240–247

[14] Cowan N (2001) The magical number 4 in short-term memory: A reconsideration of mental storage capacity. Behavioral and brain sciences 24(1):87–114

[15] De Wolf R, Pierson J (2013) Whos my audience again? understanding audience management strategies for designing privacy management technologies. Telematics and Informatics

[16] De Wolf R, Gao B, Berendt B, Pierson J (2015) Interactive grouping technology for social network sites: exploring users' perceived values and actual behaviours, submitted for publication, at the ACM conference on Computer-Supported Cooperative Work and Social Computing

[17] Dunbar R, Sutcliffe A (2012) Social complexity and intelligence. The Oxford Handbook of Comparative Evolutionary Psychology p 102

[18] Dunbar RI (1992) Neocortex size as a constraint on group size in primates. Journal of Human Evolution 22(6):469–493
[19] Fang L, Kim H, LeFevre K, Tami A (2010) A privacy recommendation wizard for users of social networking sites. In: Proceedings of the 17th ACM conference on Computer and communications security, ACM, pp 630–632
[20] Field DJ, Hayes A, Hess RF (1993) Contour integration by the human visual system: Evidence for a local association field. Vision research 33(2):173–193
[21] Fortunato S (2010) Community detection in graphs. Physics Reports 486(3):75–174
[22] Fortunato S, Barthelemy M (2007) Resolution limit in community detection. Proceedings of the National Academy of Sciences 104(1):36–41
[23] Fruchterman TM, Reingold EM (1991) Graph drawing by force-directed placement. Software: Practice and experience 21(11):1129–1164
[24] Furnas GW (1981) The fisheye view: A new look at structured files. Tech. rep., Bell Laboratories Technical Memorandum
[25] Furnas GW (1986) Generalized fisheye views, vol 17. ACM
[26] Gao B, Berendt B, Clarke D, Wolf RD, Peetz T, Pierson J, Sayaf R (2012) Interactive grouping of friends in osn: Towards online context management. In: ICDM Workshops, IEEE Computer Society, pp 555–562
[27] Ghoniem M, Fekete JD, Castagliola P (2004) A comparison of the readability of graphs using node-link and matrix-based representations. In: Information Visualization, 2004. INFOVIS 2004. IEEE Symposium on, IEEE, pp 17–24
[28] Green-Armytage P (2010) A colour alphabet and the limits of colour coding. JAIC-Journal of the International Colour Association 5
[29] Grivet S, Auber D, Domenger JP, Melancon G (2006) Bubble tree drawing algorithm. In: Computer Vision and Graphics, Springer, pp 633–641
[30] Gürses S (2010) Multilateral privacy requirements analysis in online social network services. PhD thesis, PhD thesis, Department of Computer Science, KU Leuven
[31] Healey CG, Enns JT (1999) Large datasets at a glance: Combining textures and colors in scientific visualization. Visualization and Computer Graphics, IEEE Transactions on 5(2):145–167
[32] Heer J, Shneiderman B (2012) Interactive dynamics for visual analysis. Queue 10(2):30
[33] Johnson B, Shneiderman B (1991) Tree-maps: A space-filling approach to the visualization of hierarchical information structures. In: Visualization, 1991. Visualization'91, Proceedings., IEEE Conference on, IEEE, pp 284–291
[34] Kilmer R, Kilmer WO (2014) Designing interiors. John Wiley & Sons
[35] Lee C, Cunningham P (2013) Community detection: effective evaluation on large social networks. Journal of Complex Networks p cnt012
[36] Macrae CN, Bodenhausen GV (2000) Social cognition: Thinking categorically about others. Annual review of psychology 51(1):93–120
[37] Marwick AE, boyd dm (2011) I tweet honestly, I tweet passionately: Twitter users, context collapse, and the imagined audience. New Media & Society 13(1):114–133

[38] Mazzia A, LeFevre K, Adar E (2012) The pviz comprehension tool for social network privacy settings. In: Proceedings of the Eighth Symposium on Usable Privacy and Security, ACM, p 13
[39] McAuley JJ, Leskovec J (2012) Learning to discover social circles in ego networks. In: NIPS, pp 548–556
[40] Miller GA (1956) The magical number seven, plus or minus two: some limits on our capacity for processing information. Psychological review 63(2):81
[41] Moore K, McElroy JC (2012) The influence of personality on facebook usage, wall postings, and regret. Computers in Human Behavior 28(1):267–274
[42] Moreira AA, Paula DR, Costa Filho RN, Andrade Jr JS (2006) Competitive cluster growth in complex networks. Physical Review E 73(6):065,101
[43] Munkres J (1957) Algorithms for the assignment and transportation problems. Journal of the Society for Industrial & Applied Mathematics 5(1):32–38
[44] Newman ME (2005) A measure of betweenness centrality based on random walks. Social networks 27(1):39–54
[45] Newman ME (2006) Finding community structure in networks using the eigenvectors of matrices. Physical review E 74(3):036,104
[46] Newman ME (2006) Modularity and community structure in networks. Proceedings of the National Academy of Sciences 103(23):8577–8582
[47] Nguyen QV, Huang ML (2002) A space-optimized tree visualization. In: Information Visualization, 2002. INFOVIS 2002. IEEE Symposium on, IEEE, pp 85–92
[48] Raynes-Goldie K (2010) Aliases, creeping, and wall cleaning: Understanding privacy in the age of facebook. First Monday 15(1)
[49] Rodrigues EM, Milic-Frayling N, Smith M, Shneiderman B, Hansen D (2011) Group-in-a-box layout for multi-faceted analysis of communities. In: Privacy, security, risk and trust (passat), 2011 IEEE third international conference on and 2011 ieee third international conference on social computing (socialcom), IEEE, pp 354–361
[50] Seo J, Shneiderman B (2002) Interactively exploring hierarchical clustering results [gene identification]. Computer 35(7):80–86
[51] Shannon P, Markiel A, Ozier O, Baliga NS, Wang JT, Ramage D, Amin N, Schwikowski B, Ideker T (2003) Cytoscape: a software environment for integrated models of biomolecular interaction networks. Genome research 13(11):2498–2504
[52] Shiffrin RM, Schneider W (1977) Controlled and automatic human information processing: Ii. perceptual learning, automatic attending and a general theory. Psychological review 84(2):127
[53] Shneiderman B, Dunne C (2013) Interactive network exploration to derive insights: filtering, clustering, grouping, and simplification. In: Graph Drawing, Springer, pp 2–18
[54] Shneiderman B, Wattenberg M (2001) Ordered treemap layouts. In: Proceedings of the IEEE Symposium on Information Visualization 2001, vol 73078
[55] Stevens SS (1957) On the psychophysical law. Psychological review 64(3):153

[56] Sutcliffe A, Dunbar R, Binder J, Arrow H (2012) Relationships and the social brain: Integrating psychological and evolutionary perspectives. British journal of psychology 103(2):149–168

[57] Ugander J, Karrer B, Backstrom L, Marlow C (2011) The anatomy of the facebook social graph. arXiv preprint arXiv:11114503

[58] Van Wijk JJ, Van de Wetering H (1999) Cushion treemaps: Visualization of hierarchical information. In: Information Visualization, 1999.(Info Vis' 99) Proceedings. 1999 IEEE Symposium on, IEEE, pp 73–78

[59] Wang L, Giesen J, McDonnell KT, Zolliker P, Mueller K (2008) Color design for illustrative visualization. Visualization and Computer Graphics, IEEE Transactions on 14(6):1739–1754

[60] Wang Y, Norcie G, Komanduri S, Acquisti A, Leon PG, Cranor LF (2011) I regretted the minute i pressed share: A qualitative study of regrets on facebook. In: Proceedings of the Seventh Symposium on Usable Privacy and Security, ACM, p 10

[61] Ware C, Purchase H, Colpoys L, McGill M (2002) Cognitive measurements of graph aesthetics. Information Visualization 1(2):103–110

[62] Wolfram S (2013) Data science of the facebook world. URL http://blog.stephenwolfram.com/2013/04/data-science-of-the-facebook-world/, retrieved Nov 30, 2013

[63] Yi JS, ah Kang Y, Stasko JT, Jacko JA (2007) Toward a deeper understanding of the role of interaction in information visualization. Visualization and Computer Graphics, IEEE Transactions on 13(6):1224–1231

[64] Zuckerberg M (2012) One Billion People on Facebook. URL http://newsroom.fb.com/news/2012/10/one-billion-people-on-facebook/, retrieved Jul 19, 2014